AIAA Guidance, Navigation, and
Control Conference and Exhibit
14-17 August 2000 Denver, CO

A00-37169

AIAA-2000-4473

A CONCURRENT SEQUENCING AND DECONFLICTION ALGORITHM FOR TERMINAL AREA AIR TRAFFIC CONTROL

John E. Robinson III* and Douglas R. Isaacson*
NASA Ames Research Center, Moffett Field, California

Abstract

A next-generation decision support tool for assisting terminal area air traffic controllers in the control of arrival traffic, called the Final Approach Spacing Tool (FAST), is being developed at NASA Ames Research Center in cooperation with the FAA. An earlier version of the system, called the Passive Final Approach Spacing Tool (pFAST), generated runway assignments and landing sequences. The new version of the system, called the Active Final Approach Spacing Tool (aFAST), generates heading and speed advisories by an algorithm which simultaneously sequences the aircraft and deconflicts their trajectories along each flight segment. This paper describes the concurrent sequencing and deconfliction algorithm used by aFAST to generate its arrival plan and aircraft trajectories.

Introduction

The United States' air traffic system has experienced significant growth during the past twenty years. At the nation's twenty five busiest airports, the number of operations increased steadily from 11.8 million in 1993 to 12.9 million in 1998.[1,2] This upward trend is expected to continue with more than 18.8 million operations forecast in 2015.[2] This growth has resulted in substantial increases in delays at nearly every major airport. However, environmental and geographic constraints limit the opportunities to increase system capacity by building new airports or adding new runways at existing airports.[3] Therefore, many efforts to alleviate this problem have focused on developing decision support systems to assist the air traffic management and controller workforce. These tools schedule air traffic in real-time to enable more efficient management and control of the traffic. A main objective has been to increase the human's efficiency by providing an accurate prediction of the future traffic situation. Since controller workload is a paramount concern, these systems are intended to achieve throughput benefits while not departing significantly from present-day air traffic control procedures. Reduced uncertainty of flight time estimates allows air traffic controllers to reduce excess in-trail separation while still maintaining the highest level of safety. The contraction of spacing buffers reduces the mean flight time, decreases the airborne delay, and increases airport throughput.

* Research Scientist, Terminal Area ATM Research Branch

Background

NASA Ames Research Center has been conducting research on scheduling algorithms and decision support tools for the extended terminal area for nearly twenty years. This research has resulted in the development of a suite of tools known as the Center/TRACON Automation System (CTAS).[4] One tool in particular, the Final Approach Spacing Tool (FAST), helps TRACON traffic management coordinators and air traffic controllers manage and control arrival traffic and achieve a precisely spaced flow of traffic on final approach.[5] FAST performs this function by providing landing runway assignments, landing sequences and heading and speed advisories to air traffic controllers. In order to expedite the operational deployment of FAST, initial research concentrated on a subset of the complete functionality. This core capability, known as the Passive Final Approach Spacing Tool (pFAST), encompassed only the *passive* advisories – runway assignments and landing sequences. These advisories were found to increase airport throughput and reduce arrival delay by achieving a more balanced usage of the airport's runways.[6] pFAST is currently being used operationally by the Dallas/Fort Worth TRACON to manage arrival aircraft to the Dallas/Fort Worth International Airport (DFW). The Federal Aviation Administration (FAA) has committed to deploying pFAST to several major airports as part of its Free Flight Phase One Program (FFP1).[7] Current research is focused on the development of *active* advisories – heading and speed commands. As part of the Active Final Approach Spacing Tool (aFAST), these advisories are expected to further increase airport throughput by achieving a more precise spacing of aircraft on final approach.[8] In addition, these advisories will provide a suitable mechanism for more advanced automation such as variable wake vortex separation standards. This paper describes the scheduling algorithm used by aFAST to generate its arrival plan and aircraft trajectories.

Previous Research

While developing air traffic decision support systems like FAST, researchers at various laboratories worldwide have experimented with a wide range of scheduling algorithms. Several of these approaches are documented in Reference 9. This paper limits historical discussion to the pFAST scheduling algorithm that serves as the basis for the aFAST scheduling algorithm. Unlike previous methodologies, which relied predominantly upon time-based optimization, pFAST uses trajectory-based spatial constraint satisfaction.[10] Extensive real-time simulations have shown that, to produce an acceptable arrival plan, all merges within the terminal airspace, and not simply the last merge at the runway threshold, must be considered. To accomplish this, the larger global sequencing problem of merging to the runway threshold is divided into a network of smaller local sequencing problems of merges at intersections of upstream flight segments. By repeatedly merging from upstream flight segments onto downstream flight segments, it is possible to produce a more consistent and acceptable sequence at the runway threshold. Furthermore, the consideration of individual flight segments will provide a mechanism for integrating departure aircraft with arrival aircraft during the scheduling process.[11] A refined and enhanced spatial constraint satisfaction concept remains the foundation of the present aFAST scheduling algorithm.

It is important to recognize that the primary purpose of the pFAST and aFAST scheduling algorithms is to produce an achievable, but not necessarily optimal, arrival plan. Krzeczowski showed that the delay benefits of optimal sequencing in the terminal area are small compared to the delay benefits of optimal runway allocation.[12] Furthermore, Neuman also showed that the benefits of optimal sequencing in the terminal area are small compared to the delay benefits of reduced in-trail separation buffers.[13] During operational field evaluations of pFAST, the accuracy of predicted landing sequences exceeded 85% as aircraft entered the terminal area and quickly converged to 100% as aircraft neared the final approach course.[9] While this level of pFAST sequence advisory accuracy was deemed acceptable by air traffic controllers, aFAST sequence advisory accuracy will need to be significantly better due to the nature of active advisories. Conceptually, passive advisories are the "what-to-do", while active advisories are the "how-to-do-it". In a passive advisory system, an incorrect sequencing decision only affects the sequence advisories of two aircraft. Conversely, in an active advisory system, an incorrect sequencing decision may cascade to affect the heading and speed advisories of several aircraft. More importantly, it may not be possible to determine a conflict-free solution in the presence of an incorrect sequencing decision. Since active advisories must reflect a conflict-free solution, such incorrect sequencing decisions have a much larger penalty.

Before discussing the details of the aFAST scheduling algorithm, it is useful to explain the motives for redesigning the pFAST scheduling algorithm. The pFAST scheduling algorithm implements its spatial constraint satisfaction concept by performing sequencing and conflict resolution serially.[5] First, all aircraft involved in the scheduling process are sequenced relative to each other on all flight segments. Then, each aircraft is deconflicted with the aircraft sequenced ahead of it on each flight segment. In terms of their effect on each aircraft's predicted trajectory, the impact of upstream decisions cannot be considered in future downstream decisions. Since conflict resolution occurs at the conclusion of the scheduling process, only undelayed trajectories are available throughout the sequencing process. This behavior has two significant side-effects. First, the sequencing decisions made to merge onto downstream flight segments must be made on the basis of derived criteria rather than trajectory-based criteria. For example, derived criteria include time-based parameters, like delay incurred, whereas trajectory-based criteria, like the relative time or distance of an aircraft to a particular merge point, are physical parameters. Second, sequencing decisions are made on the earliest flight segment shared by a particular aircraft pair rather than on the flight segment where a potential conflict must be resolved. In order to illustrate these limitations, two typical situations that air traffic controllers encounter are presented.

Case 1: Merging of Base/Downwind Flight Segments

Figure 1 shows the situation where three aircraft, Aircraft A, Aircraft B and Aircraft C must be merged from different base and downwind flight segments onto the same final approach course. For illustration purposes, assume that Aircraft A and Aircraft C are predicted to reach the final approach course slightly ahead of Aircraft B. Also, assume that Aircraft A has already been determined to be first in the landing sequence. As a result, both Aircraft B and Aircraft C will need to absorb some delay in order to remain behind Aircraft A. The dashed lines represent the undelayed (i.e. fastest) trajectories of each aircraft. The solid lines represent the delayed trajectories when considering the decision to sequence Aircraft A first. If it is assumed that the next aircraft to the final approach course would be next in the landing sequence, the effect of Aircraft A on this decision is pronounced. Without consideration of Aircraft A, Aircraft C reaches the final approach course ahead of Aircraft B. However, with consideration of Aircraft A, the situation is reversed. Since serial sequencing and deconfliction does not consider the direct effect of Aircraft A on the trajectories of Aircraft B and Aircraft C, pFAST resorts to using derived quantities, such as approximating the landing time of each aircraft by computing minimum separation at the runway threshold. Such derived quantities are then considered by heuristics aimed at balancing delay between aircraft or minimizing overall delay. Conversely, concurrent sequencing and deconfliction can determine the effect of Aircraft A on the trajectories of Aircraft B and Aircraft C. Therefore, aFAST can use trajectory-based techniques such as first-come-first-served (FCFS). The importance of this capability is that these trajectory-based criteria are the same criteria used by controllers in their decision making. Furthermore, more complicated interactions such as runway dependencies or secondary merges further downstream cannot be modeled accurately by derived parameters.

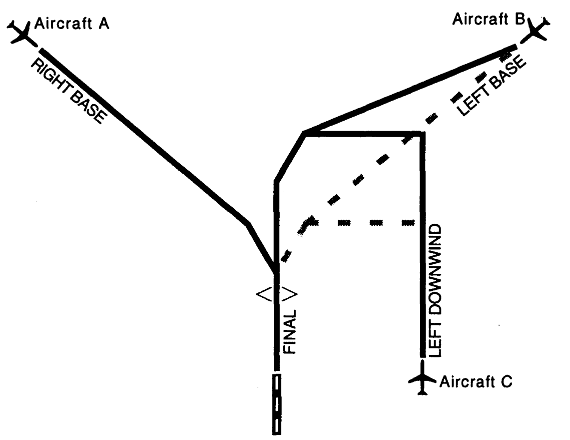

Fig. 1. Merging of Base and Downwind Flight Segments

Case 2: Overtaking on Future Flight Segments

Figure 2 shows the situation where three aircraft, Aircraft A, Aircraft B and Aircraft C must be merged from different long and over-the-top flight segments onto the same downwind flight segment. For illustration purposes, assume that Aircraft A is able to eventually overtake Aircraft B but that this overtake will not happen until both aircraft have reached the left downwind flight segment. Also, assume that Aircraft C will reach the left downwind flight segment immediately following Aircraft B, and Aircraft A will not be allowed to overtake Aircraft C. During the sequence analysis along the left long flight segment, the relative sequence of Aircraft A and Aircraft B will be evaluated first. At this point in the scheduling process, the pFAST scheduling algorithm would be unable to consider the effect of Aircraft C, namely that Aircraft A will not be able to overtake Aircraft B. Since the overtake does not happen on the left long flight segment, the algorithm can safely postpone the decision of the relative sequence of Aircraft A and Aircraft B until the left downwind flight segment is being sequenced. Specifically, the algorithm only needs to determine a relative sequence between two aircraft when that decision makes conflict resolution necessary.

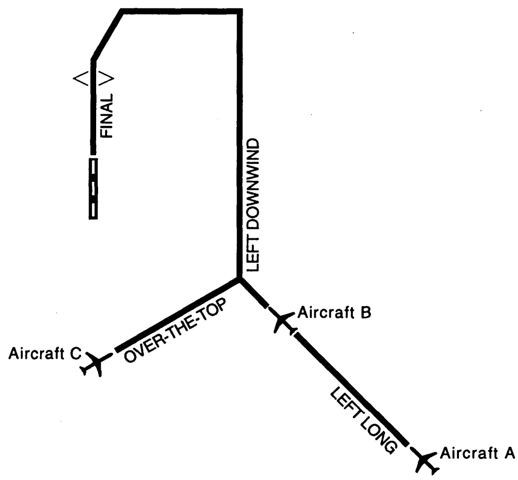

Fig. 2. Overtaking on Future Flight Segments

Scheduling Methodology

The aFAST scheduling algorithm is founded upon the same spatial constraint satisfaction concept that was developed for the pFAST scheduling algorithm. However, it must be recognized that while the concept remains the same, the aFAST scheduling algorithm represents a significant departure from the pFAST scheduling algorithm. In-depth discussions of the spatial constraint satisfaction concept are contained in References 9 and 10. A general explanation of the aFAST scheduling algorithm is described in the next section. The details of this framework are then discussed by means of step-by-step examples.

Initial Route Selection:
The scheduling process begins with the selection of an initial route for each aircraft. The initial route represents the aircraft's expected flight path without consideration for other aircraft. The selection of this route depends on a variety of factors such as the destination airport's landing configuration and the aircraft's current state. Figure 3 shows the initial route selection for Aircraft A as a solid line. The horizontal path is the shortest expected path from the aircraft's current position to the runway threshold. Each horizontal path segment is classified according to its flight segment such as downwind, base or final. The horizontal path has degrees-of-freedom which can be used to resolve potential conflicts. The initial route's airspeed profile is the fastest expected speed profile and it represents the latest locations where speed reductions can occur. The locations of these speed reductions are additional degrees-of-freedom which can also be used to resolve potential conflicts. The altitude profile approximates a standard terminal-area profile by descending along fixed flight path angles to meet a series of published altitude restrictions and standard altitude clearances.

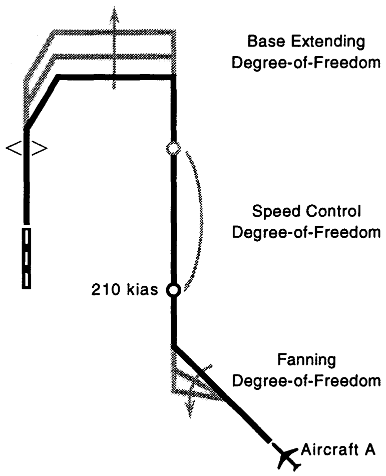

Fig. 3. Illustration of Initial Route

Trajectory Synthesis:
A necessary input to any scheduling algorithm is an estimate of the aircraft's position at a future time. The aircraft's horizontal route, airspeed profile and altitude profile are used to construct an accurate 4-dimensional (4-D) trajectory. A trajectory synthesis engine integrates point-mass aircraft equations of motion along the 3-D route to meet the desired routing, speed and altitude constraints.[14] In addition, the trajectory synthesis engine uses the winds aloft, atmospheric temperature and pressure profiles and aircraft flight characteristics. The result is a complete 4-D prediction of the aircraft's flight path from its most recently reported position to the runway threshold. This 4-D trajectory is represented discretely by a series of time steps at ten second intervals. Each time step contains the predicted aircraft state at a future time. These time steps are used by the aFAST scheduling algorithm to perform its sequencing and conflict resolution. Figure 4 shows the synthesized trajectory and discrete time steps for Aircraft A.

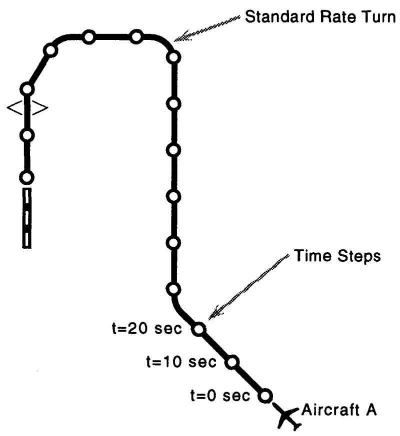

Fig. 4. Illustration of Trajectory Synthesis

Trajectory Based Sequencing:
Following each trajectory synthesis, it is necessary to define which portions of each aircraft's trajectory correspond to particular flight segments. Figure 5 shows several aircraft and their trajectories broken into typical flight segments, referred to as *left long, over-the-top, left downwind, right downwind, left base* and *final.*

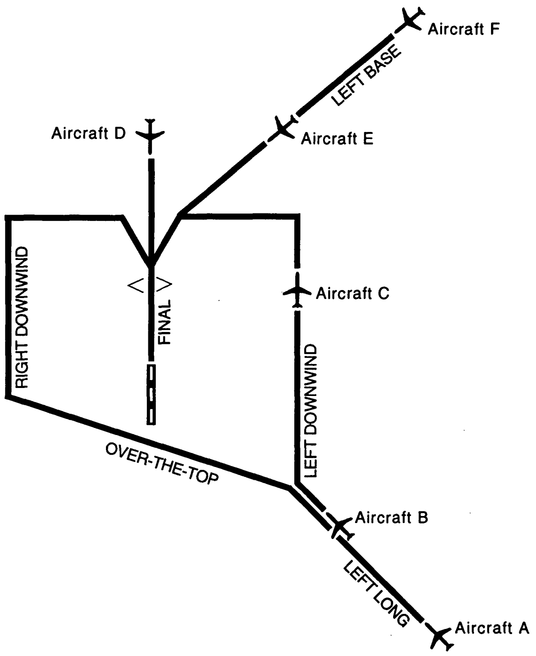

Fig. 5. Illustration of Merging Flight Segments

Relative sequences are determined (and conflict resolution performed) for all aircraft that share at least one flight segment. However, aircraft can share any number of flight segments and these flight segments do not need to be contiguous. For this reason, the scheduling process is best represented by a network of dependencies rather than a tree of sequencing decisions. For example, Aircraft E and Aircraft F share all flight segments: base and final; Aircraft A and Aircraft D share only their last flight segment: final; and Aircraft A and Aircraft B share both their first and last flight segments: long and final. Fortunately, non-contiguous flight segments are usually loosely coupled due to operational practices. Therefore, the sequencing and conflict resolution decisions between non-contiguous flight segments generally do not affect each other. This is desirable since a strong coupling between non-contiguous flight segments could cause computationally intractable circular dependencies.

The determination of the landing sequence of aircraft consists of both ordering the aircraft on each flight segment and repeatedly merging these aircraft into a consistent final sequence at the runway threshold. The flight segment network shown in Figure 6 illustrates how this sequencing problem is broken into a series of smaller sequencing problems. The arrows represent the merges that occur in the terminal airspace. The shaded boxes represent the relative sequences of aircraft which currently are on a particular flight segment and the unshaded boxes represent the relative sequences of all aircraft which eventually will be on a particular flight segment.

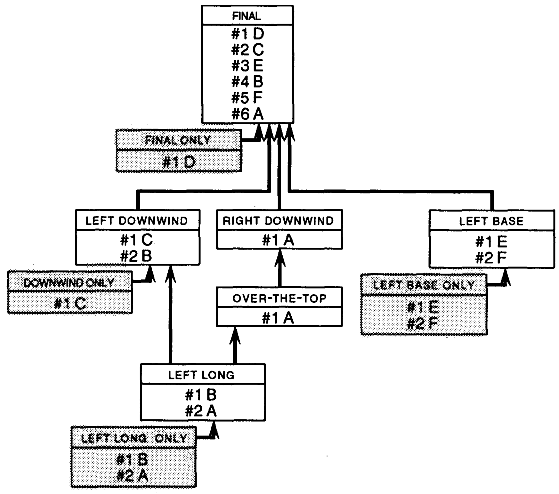

Fig. 6. Flight Segment Network

The sequencing process begins with the shaded boxes. Here, the relative sequences are determined for the set of aircraft which are currently on each flight segment. This step of the sequencing process is referred to as "ordering." A more detailed explanation of the ordering process is presented in a later section of this paper. Once the flight segments are ordered, the sequencing process continues with the unshaded boxes. Here, a relative sequence is determined by combining the aircraft currently on the flight segment with those aircraft currently on previous flight segments. This step of the sequencing process is referred to as "merging." A flight segment can only be merged after it has been ordered and all of its preceding flight segments have been merged. Again, a more detailed explanation of the merging process is presented in a later section of this paper. The ordering and merging heuristics for sequencing a particular aircraft pair consider many factors whose details are not needed to describe the trajectory-based sequencing and conflict resolution methodology. A complete description of the heuristics used by the pFAST scheduling algorithm are presented in Reference 9. While the exact heuristics used by the aFAST scheduling algorithm may be different, fuzzy reasoning is expected to remain the foundation of the decision making. The sequencing process is complete when all aircraft have been merged onto their final flight segment.

Concurrent Sequencing and Conflict Resolution:
The previous examples describing the ordering of overtakes and the merging of base and downwind traffic streams demonstrate the need for trajectories to be consistent with the constraints imposed by decisions made earlier in the sequencing process. As a result, the aFAST scheduling algorithm has been designed to perform conflict resolution immediately following each individual sequencing decision. This technique fundamentally distinguishes the aFAST scheduling algorithm from its predecessors that postponed conflict resolution until all sequencing decisions were made. The concurrent sequencing and conflict resolution concept is intuitively appealing because maintaining separation is an air traffic controller's primary responsibility. Air traffic controllers sequence streams of aircraft in a way that allows them to easily and efficiently maintain separation. In this way, concurrent sequencing and conflict resolution appropriately considers sequencing to be a result of maintaining separation rather than a cause of conflict resolution.

This approach improves the ability of the aFAST scheduling algorithm to simply and robustly determine a reasonable arrival plan for several reasons. First, every trajectory that the scheduling algorithm considers will already be properly constrained by earlier decisions. This means that at every point in the process, the system's estimate of an aircraft's arrival time will reflect the best estimate possible. Second, trajectory-based parameters can be used by the sequencing heuristics because the trajectories more accurately reflect the aircraft's expected flight path. Since controllers use trajectory-based parameters in their explanations of expected sequencing decisions, it is believed that trajectory-based parameters provide a better model of the controller's cognitive process. Therefore, using these parameters should simplify (and improve) the sequencing heuristics. Ultimately, it is expected that the heuristics can strongly favor FCFS to each individual merge location with special consideration made for grouping aircraft by weight class and avoiding the exhaustion of each aircraft's degrees-of-freedom. Lastly, the results of the simultaneous sequencing and conflict resolution will allow the scheduling algorithm to more easily recognize and repair its own undesirable decisions. Since the conflict resolution actions are specifically related to a single sequencing decision, a particular decision can be reversed if its associated conflict resolution actions are unreasonable.

The benefits of concurrent sequencing and conflict resolution do not come without a cost. It is expected that the aFAST scheduling algorithm will generate more trajectory requests during its update cycle than the pFAST scheduling algorithm does. This is significant since trajectory synthesis represents most of the computation load of the pFAST scheduling algorithm. The design approach has been to develop a sequencing scheme that is intuitively appealing and logically robust without specific consideration of the computation load. With the rapidly increasing power of computer processing, even a five-fold increase between concurrent and serial sequencing and deconfliction is expected to remain achievable in real-time. Furthermore, if the computation requirements prohibit real-time execution of the aFAST scheduling algorithm, steps to reduce this load can be made later.

Sequencing Procedure

The next section provides detailed descriptions of the ordering and merging processes used by the aFAST scheduling algorithm. These sequencing procedures are explained by following step-by-step examples through the algorithm.

Basic Ordering Concept:
The aFAST scheduling algorithm's design philosophy for ordering has been to make a reasonable first guess at an initial order and to then systemically refine that guess to arrive at an improved final order. The primary reason that a refinement approach was taken is that the underlying sequencing decisions remain based upon aircraft-to-aircraft analyses. Therefore, the progression in which these pairwise comparisons are made can be shown to have a direct impact on the ultimate decision. In order to illustrate this dependency, Figure 7 shows a situation in which three aircraft need to be ordered on a particular flight segment. Assume that Aircraft A and Aircraft B are flying in-trail at equivalent speeds. Also, assume that Aircraft C has the potential to overtake Aircraft B but it does not have the potential to overtake Aircraft A. From the air traffic controller's perspective, the decision to allow Aircraft C to overtake Aircraft B will depend entirely on whether or not there is a position available for Aircraft C in front of Aircraft B. If there is, Aircraft C will be allowed to overtake Aircraft B in order to occupy that available position. If there is not, Aircraft C will be constrained behind Aircraft B. Cognitively, the controller performs ordering outwards from the end of the flight segment, so aircraft are considered in geometric order. Thus,

aircraft in the lead have a mechanism for constraining aircraft following them.

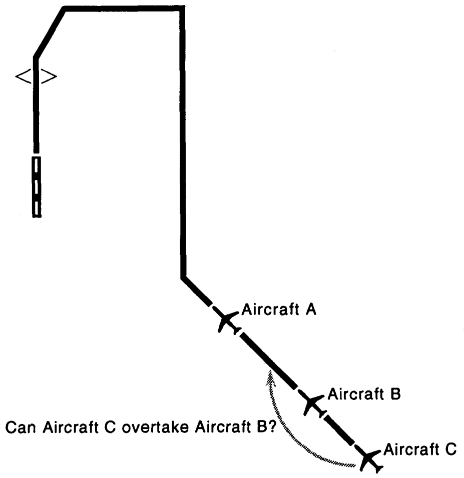

Fig. 7. Motivation for Ordering Refinement

Ordering Example:
Throughout the next several subsections, a traffic scenario illustrating the ordering process will be discussed. Figure 8 depicts a situation where four aircraft are undergoing ordering on the long flight segment. For illustration purposes, assume that the desired order is A C B D.

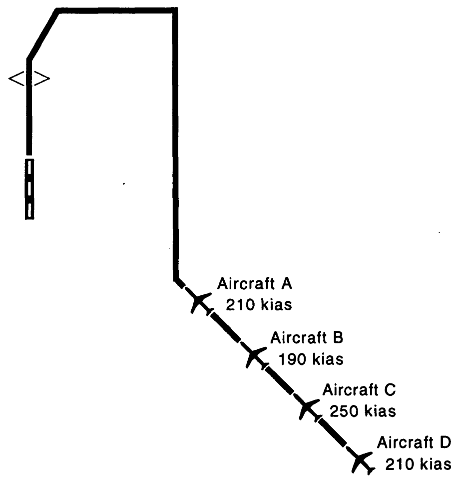

Fig. 8. Initial and Refined Ordering Example

Step 1: Initial Ordering:
The first step of the ordering process is to sequence the aircraft by their geometric distance along the flight segment. Continuing the previous example, the result of the initial ordering will be A B C D. This geometric measure is only intended to propose a reasonable first guess. Nearly all arrival aircraft destined to a busy airport, such as DFW, are eligible for and capable of meeting the same set of standard clearances issued in the terminal area. Once inside of the terminal area, traffic is generally vectored to the same target speeds and altitudes, with little regard for individual aircraft performance. Therefore, geometric distance along a particular flight segment is a reasonable measure of the eventual landing order. Occasional overtakes are permitted, so later refinements to the initial order will be made to achieve an operationally acceptable sequence.

Step 2: Refined Ordering:
The second step of the ordering process is to search the initial order for situations where refinements to the initial order are desired. Since this search operates upon the initial order, the initial order only defines the priority that an aircraft has in the determination of the refined order; it does not define the position that an aircraft has in the refined order itself. Beginning with the first aircraft, the general proposition "Can the trailing aircraft overtake the leading aircraft?" is evaluated. If the answer is yes, the order is reversed, otherwise, the order is maintained. An aircraft is not restricted in the number of so-called overtakes it can perform. However, the decision to allow such a maneuver will be evaluated one overtake at a time forward from that aircraft's position in the initial order. Once it is determined that the trailing aircraft cannot overtake the leading aircraft, conflict resolution is performed to eliminate any potential conflicts along their current flight segment. Table 1 represents this step-by-step ordering refinement for the aircraft depicted in the Figure 8 example. For a particular aircraft, the subscripts represent the other aircraft with whom it has undergone conflict resolution. Refined ordering is finished when all aircraft have undergone conflict resolution in their constrained position.

Table 1. Progression of Refined Ordering

Decision:	Result
(Initial Order)	A B C D
B is not allowed to overtake A	A B C D
B is constrained by A	A B_A C D
C is allowed to overtake B	A C B_A D
C is not allowed to overtake A	A C B_A D
C is constrained by A	A C_A B_A D
B is not allowed to overtake C	A C_A B_A D
B is constrained by A and C	A C_A B_{AC} D
D is not allowed to overtake B	A C_A B_{AC} D
D is constrained by A, C and B	A C_A B_{AC} D_{ACB}
(Refined Order)	A C B D

Basic Merging Concept:

The aFAST scheduling algorithm's design philosophy for merging has been to implement an approach that uses the relative sequences established on prior flight segments to define the priority of aircraft in the merging process. Again, the primary reason that a refinement approach was taken is that the underlying sequencing decisions remain aircraft-to-aircraft analyses. Therefore, like ordering, the progression in which these pairwise comparisons are made can be shown to have an explicit impact on the final outcome. In order to illustrate this point, Figure 9 shows the situation where four aircraft are being merged. Assume that Aircraft A and Aircraft B are currently on the flight segment being merged. Assume that Aircraft C and Aircraft D have the ability to merge into the same position between Aircraft A and Aircraft B but there is space available for only one aircraft. Aircraft C is closer to the merge location and, presumably, has less ability to be delayed behind Aircraft B. From the aircraft traffic controller's perspective, Aircraft C and Aircraft D will occupy the position that is next available to them. Cognitively, the air traffic controller performs merging outwards from the flight segment, so aircraft are considered in geometric order. Aircraft closest to the merge location have priority in the merging process (but not necessarily priority in the merging outcome). By reconciling the sequence of Aircraft C first, it becomes evident that Aircraft D must follow Aircraft B.

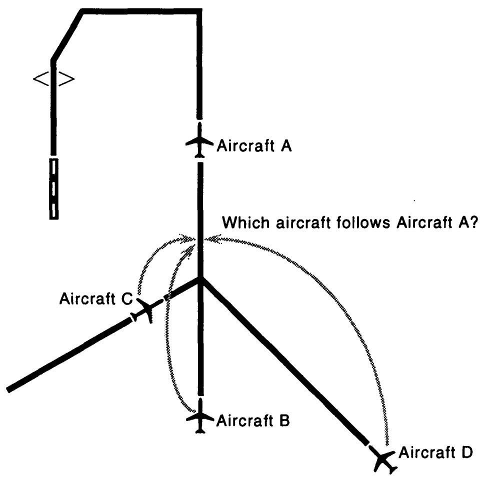

Fig. 9. Motivation for Merging Refinement

Merging Example:

The next several sections discuss an example arrival traffic scenario illustrating the merging process. Figure 10 depicts the traffic situation where seven aircraft are being merged onto a final approach course. For illustration purposes, assume the desired sequence is A P B M X. Aircraft Y and Aircraft Q are being merged onto an independent final approach course.

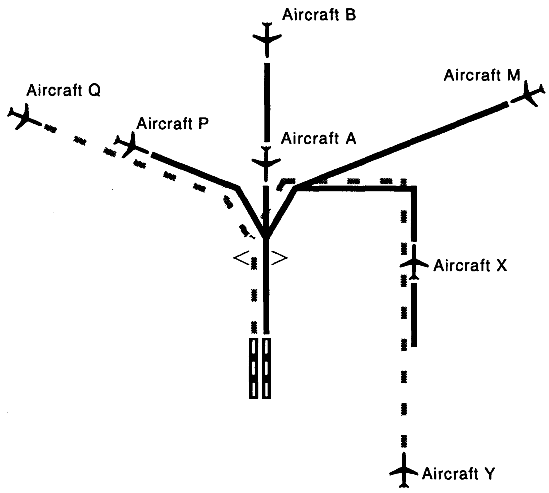

Fig. 10. Initial and Refined Merging Example

Step 3: Initial Merging:

The first step of the merging process is to determine which aircraft from the previous flight segments are involved in the merge. The aircraft explicitly involved in the merge onto final are: Aircraft A and Aircraft B from final, Aircraft M from base left, Aircraft P from base right and Aircraft X from downwind left. From the above example, Aircraft Q and Aircraft Y share flight segments with these aircraft on the downwind and base flight segments, yet they do not share the final segment with these aircraft. It is important to understand that the trajectory of Aircraft X may affect the trajectory of Aircraft Y and that the trajectory of Aircraft P may affect Aircraft Q. Therefore, Aircraft Q and Aircraft Y are implicitly involved in the merge onto final.

Step 4: Refined Merging:

The second step of the merging process is to systematically choose a single aircraft at a time from the previous flight segments and combine it with the traffic already sequenced to the flight segment being merged. During refined merging, the leading aircraft (i.e. those aircraft not yet merged) from each flight segment compete with each other for the opportunity to merge. The next aircraft to be merged is simply defined as the aircraft that is predicted to reach the flight segment ahead of its rivals in a strictly FCFS manner. This approach is intuitively appealing since air traffic controllers determine the sequence of a particular merge by scanning outward from that flight segment. This aircraft is placed amongst the aircraft already merged on the flight segment based upon the time that it first occupies that flight segment. From the previous example, at the start of merging process, Aircraft A and Aircraft B are already on the flight segment and will be considered merged. The rivals for the next opportunity to merge are the leaders of the previous flight segments, namely Aircraft M, Aircraft P and Aircraft X. Assume that Aircraft M, Aircraft P and Aircraft X are all capable of reaching the final flight segment in the position between Aircraft A and Aircraft B if unconstrained. Assume then that Aircraft P is selected since it is first to the final flight segment. Once Aircraft P has been merged with Aircraft A and Aircraft B, its position among these other aircraft is refined using the same refined ordering procedure discussed previously. Therefore, Aircraft A, Aircraft P and Aircraft B are searched for overtake possibilities. When an overtake is denied, conflict resolution is performed. Conflict resolution is performed on all flight segments already fully sequenced, so it is possible that Aircraft Q is affected by the decision to constrain Aircraft P behind Aircraft A. Following the completion of the refined ordering, the refined merging process continues. Again, the rivals for the next opportunity to merge are the leaders of the previous flight segments, namely Aircraft M, Aircraft Q (since Aircraft P has been merged) and Aircraft X. Table 2 represents this step-by-step merging refinement for the aircraft depicted in the Figure 10 merging example. For a particular aircraft, the subscripts represent the other aircraft with which it has undergone conflict resolution. Refined merging is finished when all aircraft have been merged and undergone conflict resolution in their constrained position.

Table 2. Progression of Refined Merging

Decision:	Result
(Initial Merge)	$A\ B_A$
P is next to the merge	
P is placed between A and B	$A\ P\ B_A$
P is not allowed to overtake A	$A\ P\ B_A$
P is constrained by A	$A\ P_A\ B_A$
Q is constrained by P	$P\ Q_P$
B is not allowed to overtake P	$A\ P_A\ B_A$
B is constrained by A and P	$A\ P_A\ B_{AP}$
M is next to the merge	
M is placed between P and B	$A\ P_A\ M\ B_{AP}$
M is not allowed to overtake P	$A\ P_A\ M\ B_{AP}$
M is constrained by A and P	$A\ P_A\ M_{AP}\ B_{AP}$
B is allowed to overtake M	$A\ P_A\ B_{AP}\ M_{AP}$
M is constrained by A, P and B	$A\ P_A\ B_{AP}\ M_{APB}$
X is last to the merge	
X is placed between P and B	$A\ P_A\ X\ B_{AP}\ M_{APB}$
B is allowed to overtake X	$A\ P_A\ B_{AP}\ X\ M_{APB}$
M is not allowed to overtake X	$A\ P_A\ B_{AP}\ X\ M_{APB}$
X is constrained by A, P and B	$A\ P_A\ B_{AP}\ X_{APB}\ M_{APB}$
Y is constrained by X	$X_{APB}\ Y_X$
M is allowed to overtake X	$A\ P_A\ B_{AP}\ M_{APB}\ X_{APB}$
X is constrained A, P, B and M	$A\ P_A\ B_{AP}\ M_{APB}\ X_{APBM}$
Y is constrained by X	$X_{APBM}\ Y_X$
(Refined Merge)	A P B M X

Status

Research and development of aFAST resumed two years ago following a successful field evaluation of pFAST. During this time, the general architecture of the aFAST scheduling algorithm has been designed and implemented. Determination of the specific sequencing and conflict resolution heuristics has begun. These heuristics will form the lowest level decision-making logic of the scheduling algorithm described in this paper. The current plan for aFAST development is to continue development with extensive testing of the entire aFAST scheduling algorithm.

Recently, a series of human factors evaluations were conducted with air traffic controllers from the Dallas/Fort Worth TRACON. The investigations focused on physical attributes of the active advisories, such as their symbology, use of color and presentation lead time. During these simulations, prerecorded traffic scenarios were overlayed with active advisories of various presentation formats. The active advisories were generated by offline analyses of the traffic scenarios. This approach enabled investigation of aFAST interface designs in a realistic moving-traffic environment before the aFAST scheduling algorithm was fully developed. A complete description of this novel simulation methodology and its results are described in Reference 15.

Conclusions

A scheduling algorithm for sequencing arrival air traffic in the terminal area has been developed at the NASA Ames Research Center. This algorithm will serve as the basis for the next-generation decision support system called the Final Approach Spacing Tool (FAST). The current version of FAST, known as the Passive Final Approach Spacing Tool (pFAST) provides passive advisories, namely runway assignments and landing sequences. This functionality is being extended in a new version of FAST, known as the Active Final Approach Spacing Tool (aFAST) to provide active advisories, namely heading and speed commands. As a result of information gathered during the operational testing and evaluation at the Dallas/Fort Worth TRACON, several design characteristics affecting the pFAST scheduling algorithm's sequence advisory accuracy were identified. The resulting design requirements have been used to define an improved spatial constraint satisfaction concept that will form the basis of the aFAST scheduling algorithm.

The aFAST scheduling algorithm performs trajectory-based planning by dividing the whole sequencing problem into smaller sets of short-, medium-, and long-term decisions. The scheduling process entails first ordering aircraft on their current flight segments and then repeatedly merging these aircraft onto their future flight segments. The scheduling process is completed when all aircraft have been merged onto the final flight segment. The aFAST scheduling algorithm performs sequencing and conflict resolution concurrently. Therefore, at each step in the scheduling process, the system is considering a set of aircraft trajectories which accurately reflect the spatial constraints imposed by earlier sequencing decisions. As a result of simultaneous sequencing and conflict resolution, all merging decisions throughout the terminal area will be able to consider trajectory-based parameters. It is expected that the sequencing heuristics used to model the sequencing process will more closely resemble the cognitive decisions of the air traffic controllers and, therefore, be more acceptable to controllers.

References

[1] Federal Aviation Administration. (1994). FAA Statistical Handbook of Aviation.

[2] Federal Aviation Administration. (1998). Terminal Area Forecasts – FY 1998-2015 (Report No. FAA-APO-98-10).

[3] Donohue, G. (2000). "United States and European Airport Capacity Assessment using the GMU Macroscopic Capacity Model," Proceedings of the 3rd International Air Traffic Management R&D Seminar, Napoli, Italy.

[4] Erzberger, H., Davis, T.J. and Green, S.M. (1993). "Design of Center-TRACON Automation System," Proceedings of the AGARD Guidance and Control Panel 56th Symposium on Machine Intelligence in Air Traffic Management, Berlin, Germany.

[5] Davis, T.J., Krzeczowski K.J. and Bergh, C. (1994). "The Final Approach Spacing Tool," Proceedings of the 13th IFAC Symposium on Automatic Control in Aerospace, Palo Alto, California.

[6] Davis T.J., Isaacson D.R., Robinson J.E., den Braven W., Lee K.K. and Sanford B. (1997). "Operational Test Results of the Passive Final Approach Spacing Tool," Proceedings of the 8th IFAC Symposium on Transportation Systems, Chania, Greece.

[7] Federal Aviation Administration. (1998). Government/Industry Operational Concept for the Evolution of Free Flight: Addendum 1 Free Flight Phase I.

[8] Credeur L., Capron W.R., Lohr G.W., Crawford D.J., Tang D.A. and Rodgers W.G. (1993). Final Approach Spacing Aids (FASA) Evaluation for Terminal Area Time-Based Air Traffic Control (NASA Report No. TP-3399).

[9] Robinson J.E., Davis T.J. and Isaacson D.R. (1997). "Fuzzy Reasoning-Based Sequencing of Arrival Aircraft in the Terminal Area," Proceedings of the AIAA Guidance, Navigation and Control Conference, New Orleans, Louisiana.

[10] Bergh C.P., Krzeczowski K.J. and Davis T.J. (1996). "TRACON Aircraft Arrival Planning and Optimization through Spatial Constraint Satisfaction," Air Traffic Control Quarterly, Volume 3, Number 2.

[11] Johnson C.W., Isaacson D.R. and Lee K.K. (1999). Expedite Departure Path (EDP) Operational Concept (MIT Lincoln Laboratory Report No. NASA/A-3).

[12] Krzeczowski K.J., Davis T.J. and Erzberger H. (1995). "Knowledge-Based Scheduling of Arrival Aircraft in the Terminal Area." Proceedings of the AIAA Guidance, Navigation and Control Conference, Baltimore, MD.

[13] Neuman F. and Erzberger H. (1991). Analysis of Delay Reducing and Fuel Saving Sequencing and Spacing Algorithms for Arrival Traffic. (NASA Report No. TM-103880).

[14] Erzberger H. and Tobias L. (1986). "A Time-Based Concept for Terminal Area Traffic Management," Proceedings of the 1986 AGARD Conference, No 410 on Efficient Conduct of Individual Flights and Air Traffic, Brussels, Belgium.

[15] Quinn C. and Robinson J.E. (2000). "A Human Factors Evaluation of Active Final Approach Spacing Tool Concepts." Proceedings of the 3rd International Air Traffic Management R&D Seminar, Napoli, Italy.